FREEZE_A User Guide

Author · NebulaBunny Research Lab

I. Purpose

To let anyone reproduce the public demo offline step by step — proving that “results are verifiable, signatures check out, and alignment is rigorous,” all without exposing any execution logic.

II. What’s Inside the Package

- FREEZE_A_PUBLIC.json – public manifest containing five fingerprints and key KPI summaries
- replay_min/ – minimal replay materials (desensitized)
- FREEZE_A_PUBLIC.sha256, FREEZE_A_PUBLIC.sig, cosign.pub – signature verification materials
- filelist.sha256 – complete checksum list for all files in replay_min/
- A_handbook.pdf – this user guide

Note:
The A package does not include the engine implementation — only the minimal materials required for verification.

III. Prerequisites

- A folder named FREEZE_A_PUBLIC should already be on your desktop.
(If you have a ZIP package, extract it to the desktop first.)
- Work in a fully offline environment — read only the files in the package, no private API or key access.
- Operating system: Windows 10/11 with built-in PowerShell.

IV. Steps （≤15 minutes total）

Step 1 – Unpack and Set Up

Unzip the package and move the FREEZE_A_PUBLIC folder to your desktop for easy access.

Step 2 – Check File Completeness

Run the following line in PowerShell to list files.

```
Get-ChildItem "$Env:USERPROFILE\Desktop\FREEZE_A_PUBLIC" -Recurse
```

You should see:

- FREEZE_A_PUBLIC.json
- FREEZE_A_PUBLIC.sha256
- filelist.sha256
- Folder replay_min (containing README.md, data_sample.json, replay_min.py)

Step 3 – Integrity Check

Run these two PowerShell commands in sequence and compare the “expected” and “actual” hash values.

If the SHA256 hashes match, the manifest is intact and unmodified.

```
Get-Content "$Env:USERPROFILE\Desktop\FREEZE_A_PUBLIC\FREEZE_A_PUBLIC.sha256"
certutil -hashfile "$Env:USERPROFILE\Desktop\FREEZE_A_PUBLIC\FREEZE_A_PUBLIC.json" SHA256
```

Step 4 – Structure and Fingerprint Assertions

• Create a one-click verification script named verify_A.ps1 and execute it in PowerShell.

```
Set-Content -Path "$Env:USERPROFILE\Desktop\FREEZE_A_PUBLIC\verify_A.ps1" -Value @'
$dir = "$Env:USERPROFILE\Desktop\FREEZE_A_PUBLIC"
$mf    = Join-Path $dir "FREEZE_A_PUBLIC.json"
$mfh = Join-Path $dir "FREEZE_A_PUBLIC.sha256"
$fl    = Join-Path $dir "filelist.sha256"
$base= Join-Path $dir "replay_min"

$expected=(Get-Content $mfh).Split()[0].ToLower()
$actual    =(Get-FileHash -Algorithm SHA256 $mf).Hash.ToLower()
if($actual -eq $expected){ "MANIFEST HASH: PASS" }
else{
   "MANIFEST HASH: FAIL"; "    expected: $expected"; "    actual    : $actual"
}

$ok=$true; $map=@{}
Get-Content $fl | %{
   $p = $_ -split '\s+'; if($p.Length -ge 2){
      $map[$p[-1].TrimStart('./')] = $p[0].ToLower()
   }
}
foreach($k in $map.Keys){
   $fp = Join-Path $base $k
   if(-not(Test-Path $fp)){ "MISSING    $k"; $ok=$false; continue }
   $h=(Get-FileHash -Algorithm SHA256 $fp).Hash.ToLower()
   if($h -ne $map[$k]){ "MISMATCH $k"; "    expected: $($map[$k])"; "    actual    : $h"; $ok=$false }
   else{ "OK            $k" }
}
if($ok){ "FILELIST CHECK: PASS" } else { "FILELIST CHECK: FAIL（expected）" }
'@
```

• Run the following command — you should see two PASS results displayed.

```
powershell -ExecutionPolicy Bypass -File "$Env:USERPROFILE\Desktop\FREEZE_A_PUBLIC\verify_A.ps1" |
Tee-Object "$Env:USERPROFILE\Desktop\FREEZE_A_PUBLIC\verify_log.txt"
```

Step 5 – Tamper Test

• The idea is to compare a modified hash against the precomputed one. Make a backup first.

• Modify any file inside replay_min/ (change content, add/remove files, rename paths) and re-run Step 4.

→ The filelist.sha256 check should FAIL (affects only the file list, not the manifest).

• Modify FREEZE_A_PUBLIC.json (even adding a space) and re-run Step 4.

→ The FREEZE_A_PUBLIC.sha256 check will FAIL; if signatures are used, verification will also FAIL.

• Revert the changes to their original state → the results return to PASS.

Boundary Note:

This checklist only verifies files listed in the manifest.

If you add extra files not on the list, they won't trigger an error (because no expected hash exists for them).

V. Success Criteria (All Must Be Met)

✅ Manifest hash comparison passes — unaltered.

✅ Fingerprint structure assertions pass — spec_hash is a 12-digit hex; code_git_hash is a 7–40-digit hex.

✅ Tamper test correctly triggers FAIL when the file list is modified.

VI. Delivery & Archiving

• verify_log.txt — the verification and tamper-test log.

• Screenshot of manifest PASS — proof of integrity.

• Screenshot of tamper FAIL — proof that the self-verification mechanism works.

VII. Notes

• FREEZE_A is for public demonstration and early due diligence only — zero data leakage, zero execution logic.

• If signature or assertion checks fail, review the FAQ first, then contact technical support.

Document version: v1.0 (Verification Layer — A Package)

© NebulaBunny Research Lab — All Rights Reserved.